Assignment 2
Due October 3, 2011

You may type or write your answers by hand. If you write by hand, make sure it is clearly presented. **Do not use pencils but ink**. Please put your name and student ID clearly on the submitted assignment.

You may make use of reasonable assumptions of your own for data that might be missing in the problem texts, provided that they are explicitly and clearly stated.

You may submit a partial answer to a problem. The grading will account for this.

Question 0, Feedback (1 pt extra credit)

How many hours did you spend working on this homework assignment?

Question 1, Hazards in the MIPS Integer Pipeline (4 pts)

For this exercise, assume the standard MIPS integer pipeline (illustrated in Figure A.24), but with support for a 1-cycle integer multiply (DMUL). Consider the following loop, corresponding to a FIR filter. Assume that R1 has been previously initialized to 31.

```
loop:        DSUBI R1, R1, #1
             LD    R3, 0(R2)
             LD    R4, 4(R2)
             DMUL  R5, R3, R4
             DADD  R6, R6, R5
             DADDI R2, R2, #8
             BNEZ  R1, loop
```

a. Show the timing of this instruction sequence for the MIPS pipeline *without any forwarding or bypassing hardware* but assuming a register read and a write in the same clock cycle “forwards” through the register file, as illustrated in Figure A.7. Use a pipeline timing chart like Figure A.5 or A.10. Assume that the branch is handled by flushing the pipeline. If all memory references hit in the cache, how many cycles does this loop take to execute?

b. Show the timing of this instruction sequence for the MIPS pipeline *with normal forwarding and bypassing hardware*. Again use a pipeline timing chart like Figure A.5 or A.10. Assume that the branch is handled by predicting-as-not-taken. If all memory references hit in the cache, how many cycles does this loop take to execute?

c. Assuming the MIPS pipeline with a *single-cycle delayed branch and normal bypassing and forwarding hardware*, schedule the instructions in the loop including the branch-delay slot. You may reorder instructions and modify the individual instruction operands, but do not undertake other loop transformations that change the number or opcode of the instructions in the loop. Show a pipeline timing diagram and compute the time needed in cycles to execute the entire loop.

Question 2, Hazards in the MIPS Floating Point Pipeline (4 pts)

Repeat Question 1, but this time assuming the standard MIPS floating point pipeline (illustrated in Figure A.31). Consider the following loop, slightly modified from Question 1. Again, assume that R1 has been previously initialized to 31. In the event that there is a structural hazard due to write-back contention, give the oldest (earliest) instruction priority.

```
loop:       DSUBI R1, R1, #1
            LD.D  R3, 0(R2)
            LD.D  R4, 4(R2)
            MUL.D R5, R3, R4
            ADD.D R6, R6, R5
            DADDI R2, R2, #8
            BNEZ  R1, loop
```

Question 3, Pipeline Overhead (4 pts)

Consider the example on page A-10 that compares the unpipelined and pipelined machine. Assume that the instruction mix and cycles required per instruction type in the unpipelined processor are defined as follows.

Op	Op Frequency	Op Cycles
ALU	0.4	5
Branch	0.2	2
Load	0.2	5
Store	0.2	4

For the pipelined processor, assume that each pipeline stage is perfectly balanced, and requires 1 ns in the five-stage pipeline. Also assume that the overhead for pipelining is 0.2 ns, and is fixed independently of the number of pipeline stages.

a. Plot the speedup of an ideal pipelined processor versus the unpipelined processor as the number of pipeline stages is increase from five to 25.
b. Repeat (a) for a pipelined processor with overhead as defined above.
c. Briefly discuss the implications of pipeline overhead for optimal pipeline depth.

Question 4, Branches (4 pts)

Suppose branch frequencies (as percentages of all instructions) are as follows for a particular program:

- Conditional branches: 15%
- Jumps and calls: 5%

Further, suppose that 80% of conditional branches are taken. Consider a 4-stage pipeline. Jumps and calls are resolved at the end of the second cycle. The branch target for conditional branches is also resolved at the end of the second cycle, but branch conditions aren't resolved until the end of the third cycle. Assume that the first stage of the pipeline does not modify processor state, but that subsequent stages do.

Compared with an ideal processor (with a CPI of 1), what is the speedup of the processor when:

a. The pipeline is flushed upon encountering a branch, jump or call?
b. The pipeline predicts conditional branches *are not* taken?
c. The pipeline predicts conditional branches *are* taken?

Question 5, WAW in the MIPS FP Pipeline (4 pts)

Construct a table like Figure A.21 to check for WAW hazards in the MIPS FP pipeline illustrated in Figure A.31. Do not consider integer instructions or FP divides.